

The Application of System Simulation for Engineering the Technical Computing Environment of the Lawrence Livermore National Laboratories

Kim Minuzzo
Thomas Edmunds
Lawrence Livermore National Laboratory

Larry Roche
Van Boyd
Eric Powell
Raytheon Systems Company

September 15, 1998



This is an informal report intended primarily for internal or limited external distribution. The opinions and conclusions stated are those of the author and may or may not be those of the Laboratory.

Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under Contract W-7405-ENG-48.

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

This report has been reproduced
directly from the best available copy.

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information
P.O. Box 62, Oak Ridge, TN 37831
Prices available from (615) 576-8401, FTS 626-8401

Available to the public from the
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd.,
Springfield, VA 22161

The Application of System Simulation for Engineering the Technical Computing Environment of the Lawrence Livermore National Laboratories

**Kim Minuzzo
Thomas Edmunds
Lawrence Livermore National Laboratory**

**Larry Roche
Van Boyd
Eric Powell
Raytheon Systems Company**

September 15, 1998

The Application of System Simulation for Engineering the Technical Computing Environment of the Lawrence Livermore National Laboratories

Abstract

This report summarizes an investigation performed by Lawrence Livermore National Laboratory's (LLNL) Scientific Computing & Communications Department (SCCD) and the Garland Location of Raytheon Systems Company (RSC) from April through August 1998. The study assessed the applicability and benefits of utilizing System Simulation in architecting and deploying technical computing assets at LLNL, particularly in support of the ASCI program and associated scientific computing needs. The recommendations and other reported findings reflect the consensus of the investigation team.

The investigation showed that there are potential benefits to performing component level simulation within SCCD in support of the ASCI program. To illustrate this, a modeling exercise was conducted by the study team that generated results consistent with measured operational performance. This activity demonstrated that a relatively modest effort could improve the toolset for making architectural trades and improving levels of understanding for managing operational practices. This capability to evaluate architectural trades was demonstrated by evaluating some of the productivity impacts of changing one of the design parameters of an existing file transfer system.

The use of system simulation should be tailored to the local context of resource requirements/limitations, technology plans/processes/issues, design and deployment schedule, and organizational factors. In taking these matters into account, we recommend that simulation modeling be employed within SCCD on a limited basis for targeted engineering studies, and that an overall performance engineering program be established to better equip the Systems Engineering organization to direct future architectural decisions and operational practices. The development of an end-to-end modeling capability and enterprise-level modeling system within SCCD is not warranted in view of the associated development requirements and difficulty in determining firm operational performance requirements in advance of the critical architectural decisions. These recommendations also account for key differences between the programmatic and institutional environments at LLNL and RSC.

1. Background

SCCD is engaged in a series of major upgrades to its technical computing infrastructure that will culminate in a 10 Tflop architecture during CY 2000. Important and accurate system sizing, configuration, and operations decisions should be made in advance of these upgrades to support targeted applications, many of which are in development. SCCD is seeking increased assurance for making architectural decisions and in the administration of its computing infrastructure after deployment. This need stems from the following circumstances:

- SCCD's software applications require complementary resource upgrades in addition to computational capacity (e.g., storage, networking, visualization, archiving). The resource mix must be balanced to avoid system level bottlenecks that could hamper applications performance or throughput. Although theoretical computational performance needs can be estimated from benchmarks and past performance, other resource requirements (e.g., archive sizing, interconnect bandwidths, etc.) are more difficult to size accurately without tools that can account for predicted operational workloads and system level interactions. SCCD is additionally concerned

that most of the ASCI program investment is focused on accelerating core computing technology/performance, potentially slighting other technologies (e.g., archive performance, disk I/O bandwidth, etc.) that must be deployed in concert.

- Critical performance benchmarks will be conducted during CY99. SCCD is concerned that there is currently no model for predicting benchmark performance and/or for gaining confidence that the system architecture can provide the desired level of service overall once initial benchmark results are available.
- The applications that define future computing workloads are still in development. Greater insight is required into predicted workloads and characterization of those workloads in order to properly direct the acquisition, deployment, and use of new computing resources.

1.1. Raytheon capabilities and experience

RSC has applied simulation-based performance engineering to the development of large scale computing infrastructures since 1988 including more than 30 architectural studies involving development programs totaling \$1.5 billion. These efforts have favorably impacted operational performance and system acquisition cost to a degree that overshadows the associated investment and resource requirements.

Given the similarities between some of these RSC programs and ASCI (e.g., large scale use of advanced computing technology) there are potentially performance engineering tools and processes that could bring significant value to SCCD's systems engineering program. This investigation was conducted to determine the applicability of these tools and capabilities, the practicality of their adoption within SCCD, and the associated implementation requirements.

1.2. Objectives

The study team's investigation was driven by the initial study plan which outlined the following objectives:

- 1) Develop a conceptual model of the SCCD systems environment, associated operational concepts, and planned evolution identifying candidate aspects/needs that can be effectively supported by system simulation that supports initial modeling assessment studies and capabilities planning
- 2) Conduct informal capabilities demonstrations and/or worked examples to help in determining the validity and feasibility of adopting and integrating system simulation into the SCCD systems engineering process
- 3) Define a high level plan for applying system simulation to the on-going systems engineering process and for acquiring the associated capability to perform continuing simulation-based systems engineering.

These high level objectives were re-articulated at the initial study planning meeting as follows:

- a) Determine the range of benefits based on levels of pursuit (i.e., understand what a simulation-based methodology can accomplish) (related to items 1 and 2 above) (see sections 2 and 4)
- b) Determine if SCCD has sufficient data to make simulation modeling-based performance engineering practical and determine if the SCCD technical implementation is conducive to simulation (related to item 2 above) (see Appendices A and B and sections 2.1 and 2.2)

- c) Define the proper role of system simulation within SCCD via a worked example (derived from item 2 above) (see Appendices A and B and section 3)
- d) Develop a rational approach to implementing recommendations resulting from (c) above (e.g., pilot project, etc.) (same as item 3 above and discussed fully in sections 2 and 3)

1.3. Level of Effort

The associated Time and Materials contract supported 330 total hours of RSC effort when allocated to the following skill-sets:

Performance Engineering:	158 hrs
Simulation tools expert:	140 hrs
Management and administrative	32 hrs

SCCD provided complementary technical and management support for the collection, analysis, presentation, and interpretation of background material and performance data and co-authored this report with RSC.

1.4. Method

The investigation was performed through collaborative analysis of SCCD resource planning and systems architecting needs along with a *worked example* in which a high level simulation of a subset of the current SCCD Problem Solving Environment (PSE) was developed, exercised, and validated. The worked example served to focus the investigation and provided SCCD with insight into the simulation process, tools capabilities, resource requirements, and its potential value.

Table 1-1 identifies the core investigative team and their affiliations

Table 1-1: The Core Investigative Team

INDIVIDUAL	AFFILIATION	ROLE	CONTACT INFO
Kim Minuzzo	LLNL	SCCD system architect and study lead	925.422.2141
George Richmond	LLNL	NSL-Unitree specialist	925.423-9833
Tom Edmunds	LLNL	Statistician and performance analyst	925.422.5156
Van Boyd	RSC-G	Performance engineering technical lead	972.205.4265
Eric Powell	RSC-G	System simulation tools and modeling specialist	972.205.5603
Larry Roche	RSC-G	RSC Project manager	972.205.5641

Technical interchange to support the investigation was conducted throughout the study period in the form of meetings (site visits), E-Mail, phone calls, and technical inputs for the worked example. Specific noteworthy events include:

1. Initial technical interchange meeting (April 7-8 at LLNL)--Developed common understanding of investigative context, baselined the worked example, and defined schedule events for worked example development

2. Strategic planning meeting (August 20 at RSC)--Assessed simulation results and formulated adoption strategy and report.

2. Study Findings and Recommendations

2.1. Tools Applicability

The worked example validated the technical feasibility of bringing simulation-based performance engineering into the ASCI context (see Appendices A and B). This activity developed a first order model of SCCD's NSL-Unitree system to assess model performance and tools.

To develop the model, RSC utilized its Open Architecture Modeling environment (OAM) consisting of several commercial products and specialized reusable models and tools that have been developed by RSC to support its large systems development programs. SCCD-provided logs of user requests from the operational system served as input. The model was instrumented to report corresponding wait-times and other parameters that were then compared with operational performance logs. The model results correlate to the live log data provided by SCCD within expectations. Noted differences are attributed to several simplifying assumptions, which could easily be addressed in future model upgrades that are less cost-constrained if desired.

The model itself was developed and exercised with approximately one person-month of effort by RSC, due in part to having the tools and processes in place to support model-based studies. It is estimated that it would have taken SCCD 16-18 person-months to create a similar model using COTS modeling tools. Considerably more effort is required to develop a system level simulation of the entire PSE and to achieve the fidelity and confidence necessary for impacting configuration decisions and other aspects of the system. This appears to be beyond SCCD's current staffing resources which are not sized to support a major system modeling initiative or indoctrinated in the use of simulation in its processes.

In addition to proving technical feasibility, the worked example has drawn attention to some of the practicalities of simulation-based performance engineering that should be addressed by the implementation plan. Particularly noteworthy is the amount of support required from domain specialists to provide the necessary data collection and fact finding support for model development as well as the local effort required to drive the process and massage performance data for use in the simulation. This task carries value of its own, however, in that it stimulates discovery and understanding of system performance and usage that might otherwise not happen.

In summary, simulation modeling has applicability in the context of a structured performance engineering program but a significant commitment is needed throughout the organization to support the establishment and use of these capabilities. It helps to have a validated tool set as a starting point but the knowledge for using a tool-set can only be developed over time and is less of a concern than developing the resources required to design and conduct the modeling studies themselves.

2.2. Process Considerations

RSC programs typically deal with architecture development, deployment/transition, and operations. These development programs generally utilize future technology and are subject to highly structured and non-relaxable operational requirements from the outset. These factors make full-scale simulation-based performance engineering imperative as the tool that most closely approximates the system being designed. Simulation based studies are also used to set performance requirements for applications so that the delivered integrated system of hardware and software is compliant with its operational requirements.

After deployment, system-level simulation becomes a support-tool as the performance engineering practice shifts to the analysis of live operation. The understandings gained in the simulation phase are invaluable in the interpretation and use of live data in managing the system. Simulation studies still come into play after deployment to support architectural trades, to forecast the impacts of future workloads, and to plan and design the associated upgrades.

The circumstances within SCCD would most likely limit the value that a full-scale system level simulation could provide:

- SCCD does not have a set of firm operational requirements against which to evaluate system level performance predictions and make concrete architectural decisions

- SCCD is not set-up to flow performance requirements into algorithm implementations to ensure that performance targets for individual codes are achieved
- There is uncertainty in the performance demands that the algorithms under development will make on the computing assets
- It is unclear how simulation results would be incorporated into the current SCCD architectural decision and design process

These factors do not diminish the value that can be obtained from the targeted use of simulation in upgrade planning and resource management. For example, simulation-based studies of HPSS would have the benefits of developing core understanding of HPSS operation, its limitations, performance expectations, and in setting administrative direction.

This capability would also be very helpful in systems management. Typically this involves a Capacity Planning process that collects performance metrics on existing assets and determines future sizing and architectural requirements based on observed trends and forecasted changes in usage. It provides performance statistics that characterize current/planned workloads needed for simulation-based studies and for resource planning/management. The Resource Management process defines and implements policies for resource usage that support the capacity plans. RSC has found that simulation models developed to support system design decisions subsequently serve as trusted support tools for making capacity planning and resource management decisions/plans.

2.3. Personnel Considerations

SCCD has specialists who are focused on the evolution and operation of the various system resources but the expertise for overall architectural engineering is somewhat diluted by many responsibilities. Thus a program that overlays performance engineering responsibilities on the existing team must address the associated staffing requirements and critical skills. Individuals must be allowed to focus on the art/science of performance engineering without the burden of other day-to-day fire-drills and critical issues currently being covered by the existing team.

To implement an effective performance engineering function of this magnitude requires at least three dedicated individuals to report through the systems engineering function. These individuals should have specialized but overlapping skills in the areas of system architecture analysis and design, simulation modeling, and capacity measurement, testing, and planning.

3. Implementation Recommendations

RSC's program has been in development for more than 10 years and was accelerated by extreme economic and political pressure for accurate performance prediction. Implementing such a program still requires significant time/patience even when the target process is clearly understood. For SCCD the team recommends an evolutionary approach in which personnel, process, and tools are grown on a timetable that reflects the practical realities of identifying and acquiring good people and developing the core critical skills.

In the mean time, nothing prevents SCCD from defining its long term direction in this area while addressing acute concerns with targeted studies that fit its current resources and skill-set while setting the associated staffing and process changes in motion. The potential role of RSC and its tools in this process is a matter for further discussion.

4. Appendices A and B--Worked Example Development and Application

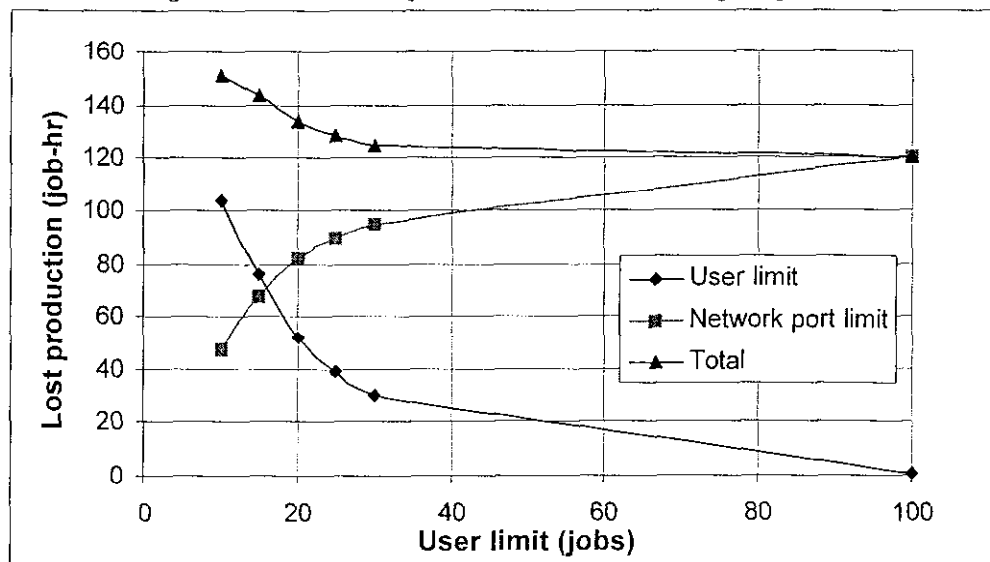
RSC and SCCD collaborated on the design and implementation of a worked example. The worked example modeled the existing PSE at a high level, focusing on archival transactions and performance for the NSL-Unitree system. The RSC system simulation tool-set was utilized by RSC modeling specialists to reduce the time required for actually implementing the simulation(s). The model was driven by scenario data collected from system logs by SCCD. The model was exercised and predicted performance statistically close to the collected validation data. The file transfer data and simulation model design and results are presented in Appendices A and B, respectively. To illustrate how the simulation model can be used to support design decisions, the productivity impacts of changing one of the design parameters of the NSL-Unitree system were examined.

The NSL-Unitree configuration currently limits each user to at most 10 active jobs. Domain experts indicated that the NSL-Unitree system would fail if the number of active jobs per user were increased to a higher value. However, if a software or hardware change could be implemented to increase the number of active jobs, a productivity gain would be realized.

To quantify the productivity impacts of this design change, the simulation model was run with a series of values for this design parameter. System performance was measured in terms of lost production by multiplying the average queue length for jobs that were waiting because of the user limit by the average time the jobs waited. The result is the total lost production, measured in job-hours. Results for a two week simulation are shown in the following figure.

The data indicate that if the user limit were increased from 10 to 20 jobs per user, the lost production due to this limitation would decrease from 100 job-hours to 50 job-hours. However, the network port limitations associated with the Unitree would increase lost production from 50 job-hours to 80 job-hours. The top curve in the figure indicates that the net effect of this design change is to reduce lost production by approximately 20 job-hours for this two week period.

Figure 4-1 Productivity loss vs. number of active jobs per user



If a dollar cost were assigned to delayed jobs, then this dollar cost could be compared to the cost of design changes that would be needed to increase the user job limit. Similar studies could be performed for other system parameters such as SCSI rate, tape size, disk size, and purge algorithm characteristics. Moreover, if the same delayed job cost were used for all of the trade studies, resources could be optimally allocated among competing design proposals.

Appendix A – Unitree File Transfer System Statistics

1.0 Background

The Problem Solving Environment (PSE) group at LLNL has been charged with developing the infrastructure to support operation of a 3 Teraflop ASCI computer in FY99 and a 10 teraflop computer in FY00. It is anticipated that availability of this larger computation capability will lead to at least a 1000-fold increase in demand for file transfer resources at LLNL.

Various configurations of hardware, software, and operating policies could be used to accommodate this large increase in demand on the file transfer system. To help evaluate candidate configurations, a discrete event simulation model is being developed. Using this model, the performance of various configurations could be estimated in order to identify an optimal configuration.

The first stage of development of this modeling capability is to build a model of the exiting Unitree disk and tape servers. Using file logs, a trace of file storage and retrieval requests will be generated to drive the simulation model. Results of the simulation model will be compared with actual file transfer times in order to validate the model. The model will then be modified to represent new hardware, software, and operating policies, and to evaluate system performance under anticipated file transfer loads.

This report describes the file transfer data that are used to estimate hardware performance characteristics and to generate the trace that will be used to drive the simulation model.

2.0 File transfer times

File transfer times from the Unitree disk cache to users and from users to the disk cache were taken from NFT, FTP daemon, and Endeavor logs from October 1, 1997 through April 9, 1998. The data source and the fields in the raw data files are described in Attachment A.

The log data include all NFT and FTP transfers. The NFT transfers were recorded to the nearest millisecond. However, the transfer times for FTP transfers were reported to the nearest second, with all subsecond transfer times estimated to be 0.499 sec. Because of this rounding, the FTP transfer times for smaller files were deemed to be unreliable. To eliminate these data, a filter was implemented to remove all transfers with times equal to 0.499 sec, and all integer file transfer times that were less than 10 seconds. Note that a negligible amount of the NFT data were discarded by this process (odds of an exact integer time for NFT are 1 in 1,000).

The data for the week of April 3 through April 9 were filtered and transfer statistics were compiled. Results are shown in Table 2.1. The data indicate that 73% of the records in the original data set were retained, which corresponded to 98% of the GB transferred. The larger

fraction of GB retained indicates that the filtering process biased the data set towards the larger file sizes.

Table 2.1 File Transfer statistics for April 3 through April 9

Date	All NFT + all FTP transfers		All NFT + FTP > 10 sec transfers			
	# transfers	GB transferred	# transfers	GB transferred	% of transfers	% of GB
980403	3,857	39.1	3,132	38.3	81%	98%
980404	927	13	763	13	82%	100%
980405	1,004	11.2	798	11.2	79%	100%
980406	2,980	39.7	1,791	38.9	60%	98%
980407	3,126	53.1	2,124	51.9	68%	98%
980408	4,015	51.4	2,848	49.7	71%	97%
980409	3,516	49.6	2,809	48.1	80%	97%
Total	19,425	257.1	14,265	251.1	73%	98%

File transfer times as a function of file size are shown in Figures 2.1 and 2.2. (Figure 2.2 is a blow up of a portion of Figure 2.1). As indicated by the data, file transfer times can vary

Figure 2.1 File transfer times vs. file size (0 – 2 GB)

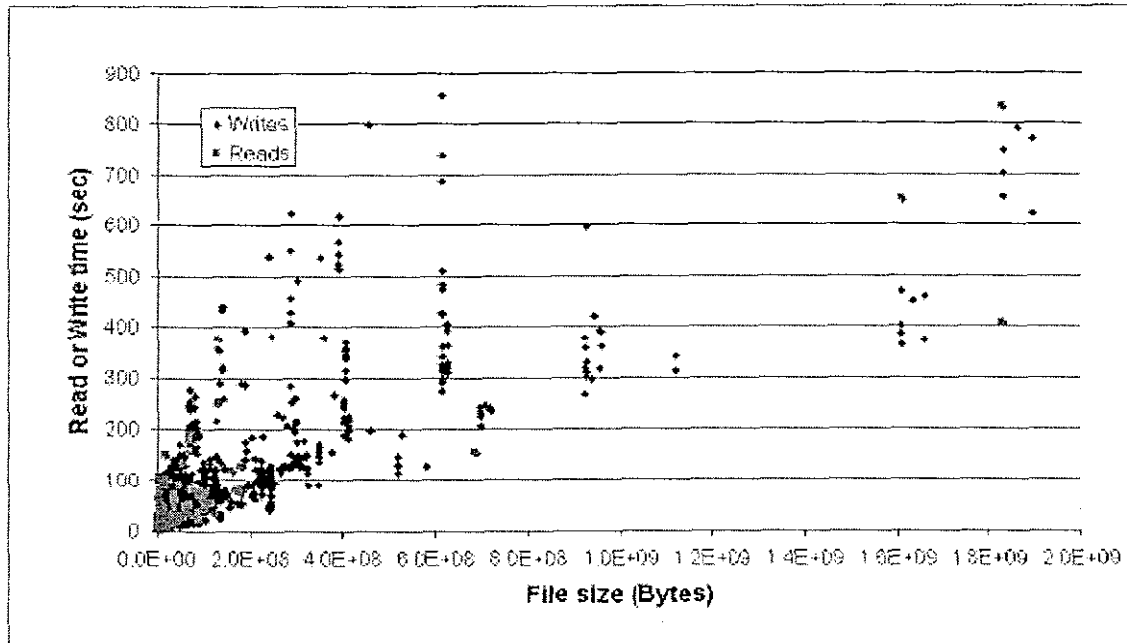
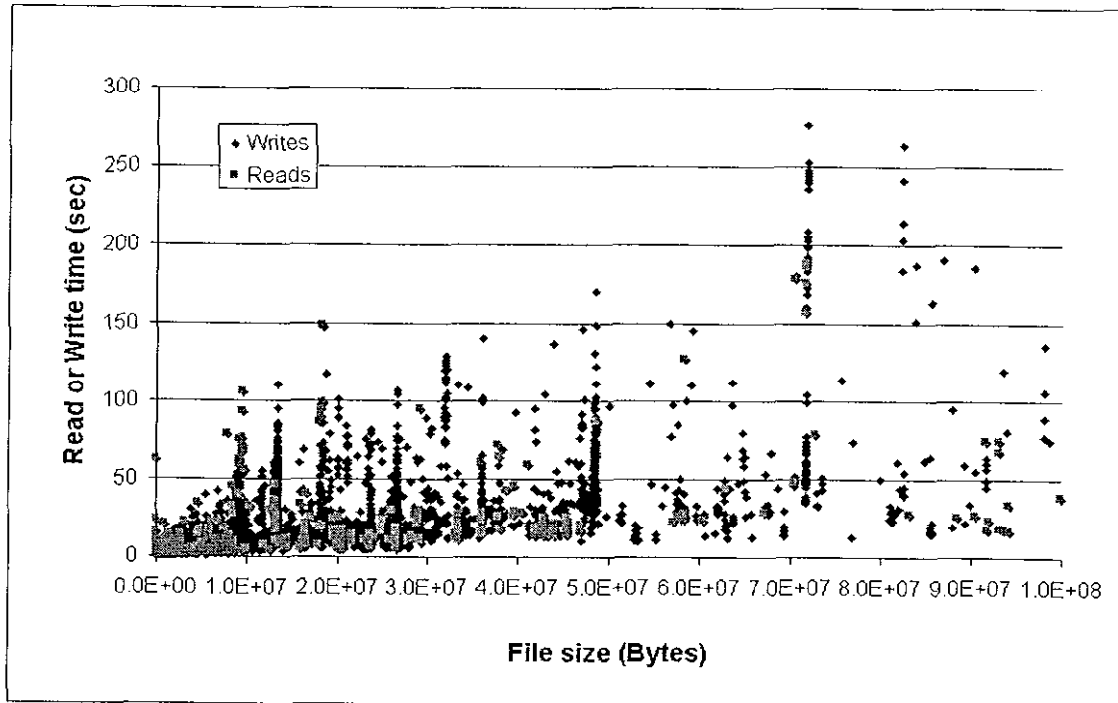


Figure 2.2 File transfer times vs. file size (0 – 100 MB)



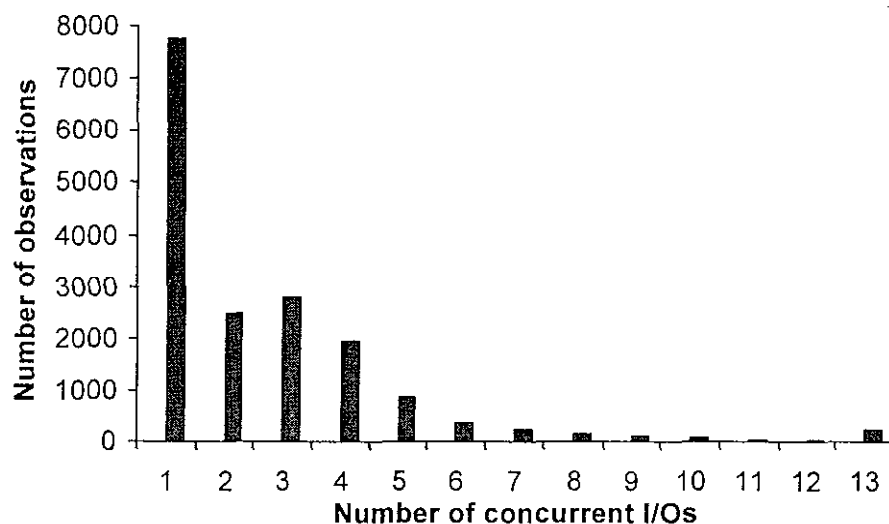
significantly for the same file size. For example, the data in Figure 2.1 show that a 600 MB file can take between 250 and 850 seconds to transfer. More severe variation is shown by the data in Figure 2.2 for 75 MB files. No systematic dependence between file transfer time and the compute platform (DEC Apha, IBM SP2, Cray YMP, etc.) was observed.

3.0 Contention for Network resources and Unitree ports

Two possible sources of the wide variation in transfer time are network contention and contention for ports to the Unitree disk cache. To investigate this, the time required for file transfer and the timestamp at the end of the transfer were used to estimate the maximum number of concurrent file transfers. This number can then be compared to network and Unitree port limits. The concurrency data are shown in Figure 3.1.

As shown by the data in the figure, the maximum number of concurrent file transfers for this week was thirteen. The network can accommodate in excess of 25 concurrent transfers, so it is unlikely that network limitations are delaying file transfers.

Figure 3.1 Concurrent file transfers



The Unitree system has thirteen ports. This constraint is reached for 213 of the 17,000 file transfers shown in Figure 3.1 (1.3%). Any time the number of concurrent transfers reaches 13, the transfer is delayed by Unitree limitations.

Contention for the Unitree ports can also be a factor when the number of concurrent transfers is less than 13. For example, the large variation in transfer times for the series of ~600 MB file transfers shown in Figure 2.1 is due to contention for Unitree ports 12 and 13. The Unitree port allocation logic assigns files of size greater than 256 MB to ports 12 and 13. If these ports are busy, the files must wait for transfer. Some of the data for these 600 MB file transfers are shown below in Table 3.1

Table 3.1 600 MB file transfer data

Index	User	Date	Time	Transfer	Bytes	Seconds	Bytes/sec	Concurrent
1	1034	98/04/05	0:45:06	si	12,345	0.182	68	1
2	1034	98/04/05	0:45:08	is	12,345	0.575	21	1
3	901	98/04/05	0:45:13	js	404,389,888	343.201	1,178	4
4	901	98/04/05	0:46:57	js	408,911,872	204.867	1,996	4
5	901	98/04/05	0:52:49	js	624,132,096	309.188	2,019	1
6	5151	98/04/05	0:55:40	sj	12,345	0.169	73	1
7	5151	98/04/05	0:55:42	js	12,345	0.723	17	1
8	901	98/04/05	0:57:46	js	612,958,208	295.939	2,071	3
9	901	98/04/05	1:01:42	js	613,318,656	855.679	717	5

As indicated by the data in the table, transfer 9 is completed at 1:01:42 and took 856 seconds (approximately 15 minutes). The data in the last column of the table indicate that there were four other transfers going on concurrently (transfers 5, 6, 7, and 8). Two of these transfers, 5 and 8, would have been assigned to ports 12 and 13. Hence, the ports that could potentially be assigned to transfer 9 were unavailable and caused delay of the transfer. Similar patterns were observed for the ~80 MB file transfers shown in Figure 2.1. Contention occurs because the five ports, numbers 9 through 13, are allocated to files greater than 4 MB and requests for these ports exceed their availability. Contention for Unitree ports will be modeled explicitly in the simulation model.

4.0 Latency and transfer speed

The transfer times shown in Figures 2.1 and 2.2 reflect contention for Unitree ports. However, the simulation model must incorporate hardware performance characteristics such as latency and transfer speeds in order to estimate contention for resources due to a given series of requests for file transfers (a trace).

To estimate hardware performance, points representing the lower envelop of the data shown in Figure 2.1 were extracted, and a regression line was fit to the data. The following procedure was used. Results are shown in Figure 4.1, Table 4.1, and Table 4.2.

- 1) Use file transfer data from 4/2/98 to 4/9/98
- 2) Remove small FTP transfers (all transfers with transfer times equal 0.499 sec, or integer time that is < 10 sec)
- 3) Select and sort write transfers in ascending order with respect to file size
- 4) Remove transfers with 0 transfer time
- 5) Stratify transfers with 100 transfers in each stratum
- 6) Choose fastest transfer from each stratum
- 7) Fit regression line to writes
- 8) Repeat steps 3) through 7) for reads, assign 25 transfers to each stratum

The data in Figure 4.1 indicate that the transfer rate for writes and reads, the reciprocal of the slopes of the lines, are approximately equal. However, the latency for writes, the intercept on the vertical axis, is larger than that for reads. The information in Tables 4.1 and 4.2 indicate that the regression models for writes and reads are statistically valid. Results are summarized in Table 4.3.

Figure 4.1 Lower envelope of file transfer times and regression fit

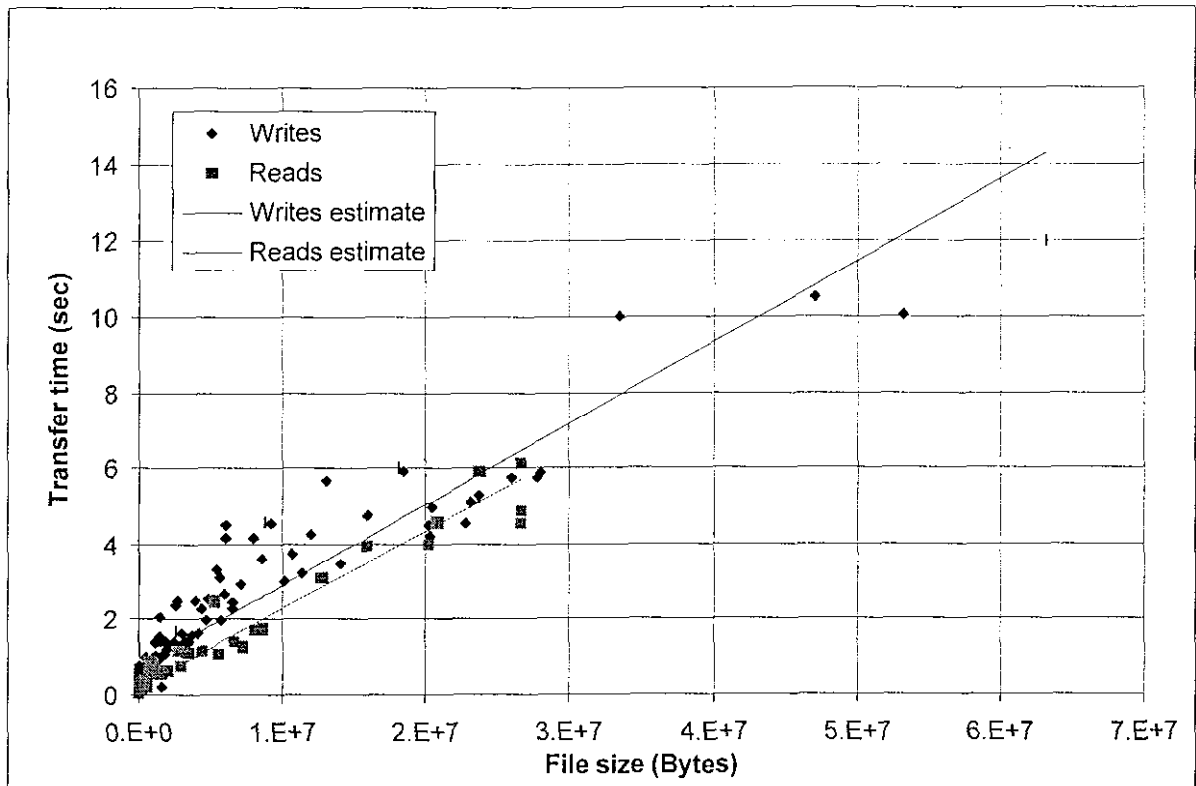


Table 4.1 Regression statistics for lower envelope of writes

Regression Statistics						
Multiple R	0.953703714					
R Square	0.909550773					
Adjusted R Square	0.908999254					
Standard Error	0.684348244					
Observations	166					
ANOVA						
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>	<i>Significance F</i>	
Regression	1	772.36085	772.3609	1649.172	1.73533E-87	
Residual	164	76.80653322	0.468333			
Total	165	849.1673832				
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>	<i>Lower 95%</i>	<i>Upper 95%</i>

Intercept	0.75603448	0.058893727	12.83727	1.48E-26	0.639746718	0.872322
	5					3
X Variable 1	2.14378E-07	5.27894E-09	40.61	1.74E-87	2.03954E-07	2.248E-07
Latency (s)	0.75603448					
	5					
Transfer rate(MB/s)	4.66466208					
	9					

Table 4.2 Regression statistics for lower envelope of reads

Regression Statistics						
Multiple R	0.978647215					
R Square	0.957750372					
Adjusted R Square	0.957241341					
Standard Error	0.307235671					
Observations	85					
ANOVA						
	df	SS	MS	F	Significance F	
Regression	1	177.6032099	177.6032	1881.514	8.35394E-59	
Residual	83	7.834681879	0.094394			
Total	84	185.4378918				
	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%
Intercept	0.204141142	0.036692423	5.563578	3.15E-07	0.131161368	0.2771209
X Variable 1	2.07143E-07	4.77548E-09	43.37643	8.35E-59	1.97645E-07	2.166E-07
Latency (s)	0.204141142					
Transfer rate(MB/s)	4.827573392					

Table 4.3 Summary of Latency and transfer rate analysis

Latency (seconds)

Transfer rate (MB/second)

	Mean	Lower 95%	Upper 95%	Mean	Lower 95%	Upper 95%
Write	0.76	0.64	0.87	4.7	4.4	4.9
Read	0.20	0.13	0.28	4.8	4.6	5.1

5.0 Residuals

The system simulation model should exhibit the same general behavior shown in Figures 1.1 and 1.2. The performance of the hardware was estimated in the previous section. Delays due to resource contention can be estimated by subtracting this ideal transfer time from the actual time to obtain an residual error that is due to contention.

Due to the small number of large files in the data set shown in Figures 1.1 and 1.2, the data set was augmented by adding all file transfers greater than 50 MB that occurred in the month of March 1998. Several large file transfers appeared to be outliers in this expanded data set. Accordingly, all files greater than 4 GB were removed. Several smaller files also appeared to be outliers, so files smaller than 1 GB that took in excess of 1000 seconds to transfer were also removed from the data. The data in Figures 5.1 and 5.2 display the file write data after these outliers were discarded. The data set includes over 20,000 file transfers.

Figure 5.1 Residual write times vs. file size

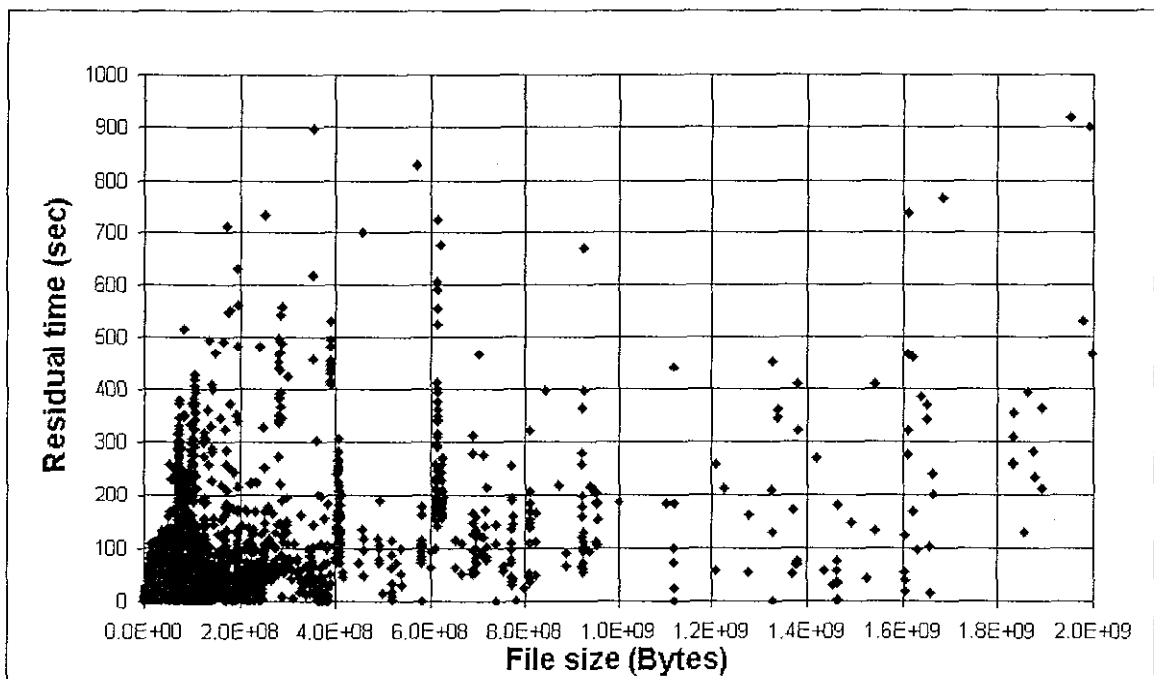
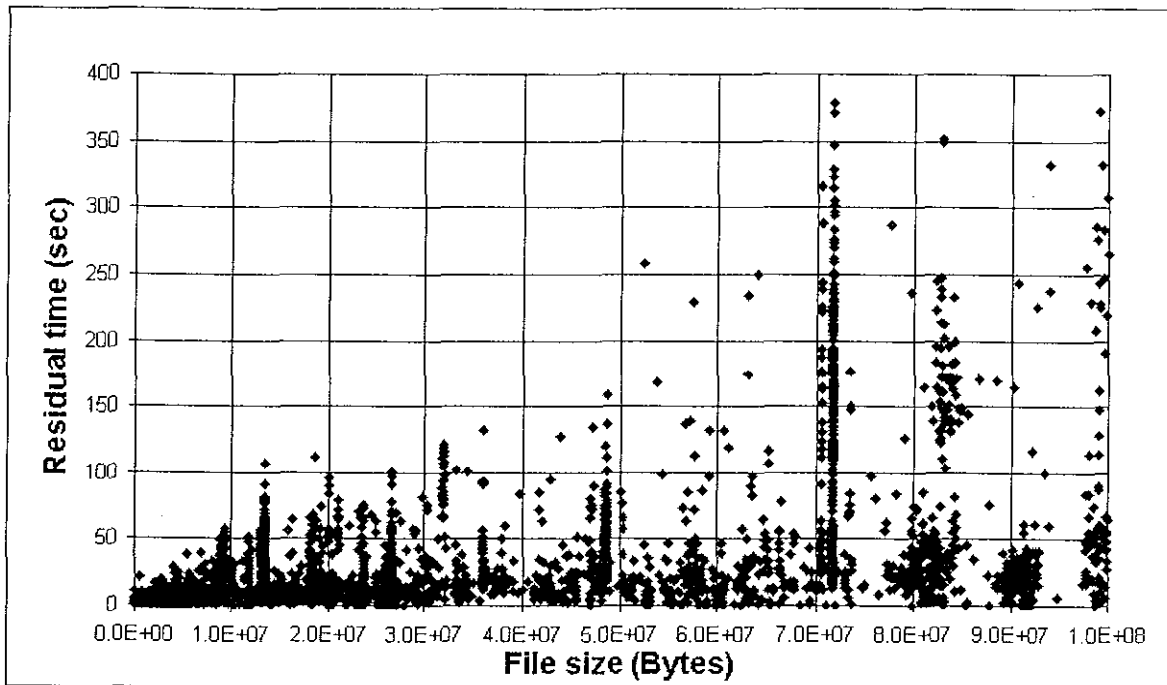
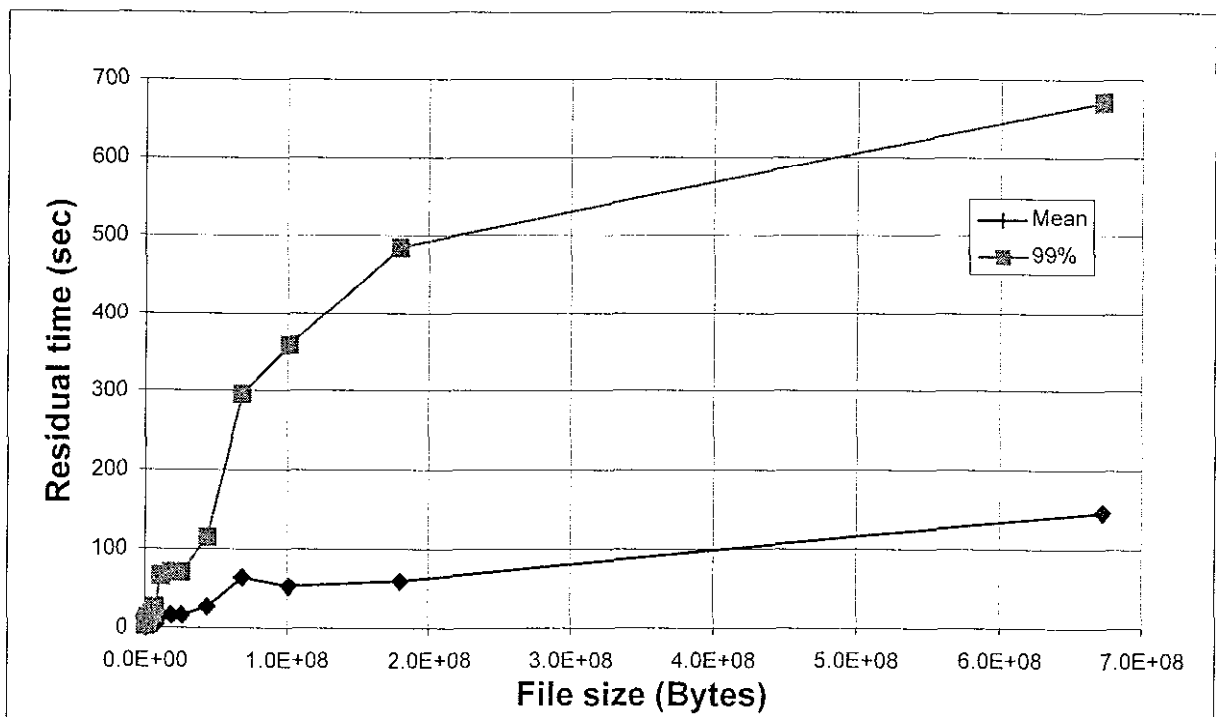


Figure 5.2 Residual write times vs. files size (blow up of 0-100 MB)



The data in Figure 5.2 indicate that the residual time increases with increasing file size. To characterize this behavior, file transfers were grouped into blocks of 1000 transfers, and the mean and the 99th percentile for each block were calculated. Results are shown below in Figure 5.3.

Figure 5.3 Mean and 99th percentile for blocks of 1000 writes



The simulation model under development should reproduce this general behavior. For example, the estimated transfer time for 179 MB file writes can be estimated using the latency and transfer rates estimated in the previous section in conjunction with the residual time data shown in Figure 5.3. The data in Table 4.3 indicate that the latency for a file write operation is 0.76 seconds, and the file transfer rate is 4.7 MB/sec. This implies that the time required to write a 179 MB file would be $0.76 + 179/4.7 = 38.8$ seconds if there were no resource contention in the system. However, the data shown in Figure 5.3 indicate that, on average, resource contention will add approximately 60.8 seconds to the file transfer time to yield a total of 99.6 seconds. Moreover, the 99th percentile curve in the figure indicates that 1% of the file 179 MB transfers would experience a delay of more than 484.6 seconds due to resource contention.

The residual delay times for file read operations are shown in Figure 5.4. The mean and 99th percentile for blocks of 200 read operations are shown in Figure 5.5. Statistics for read operations generated by the simulation model should replicate this behavior.

Figure 5.4 Residual read times vs. file size

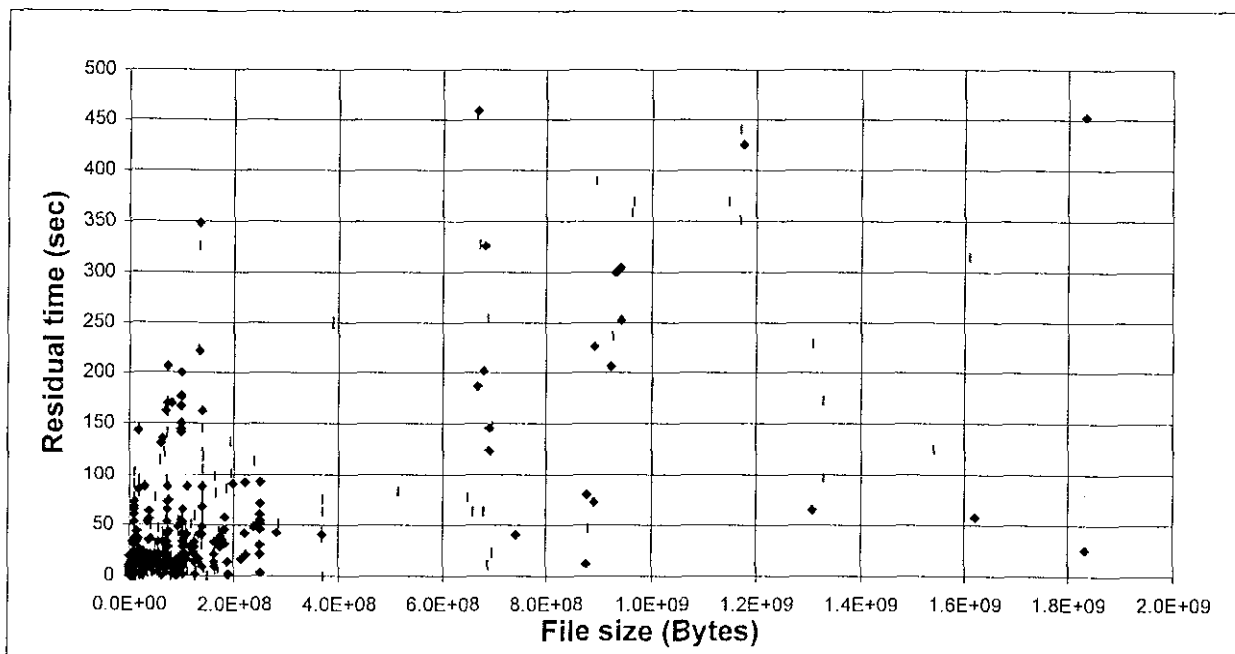
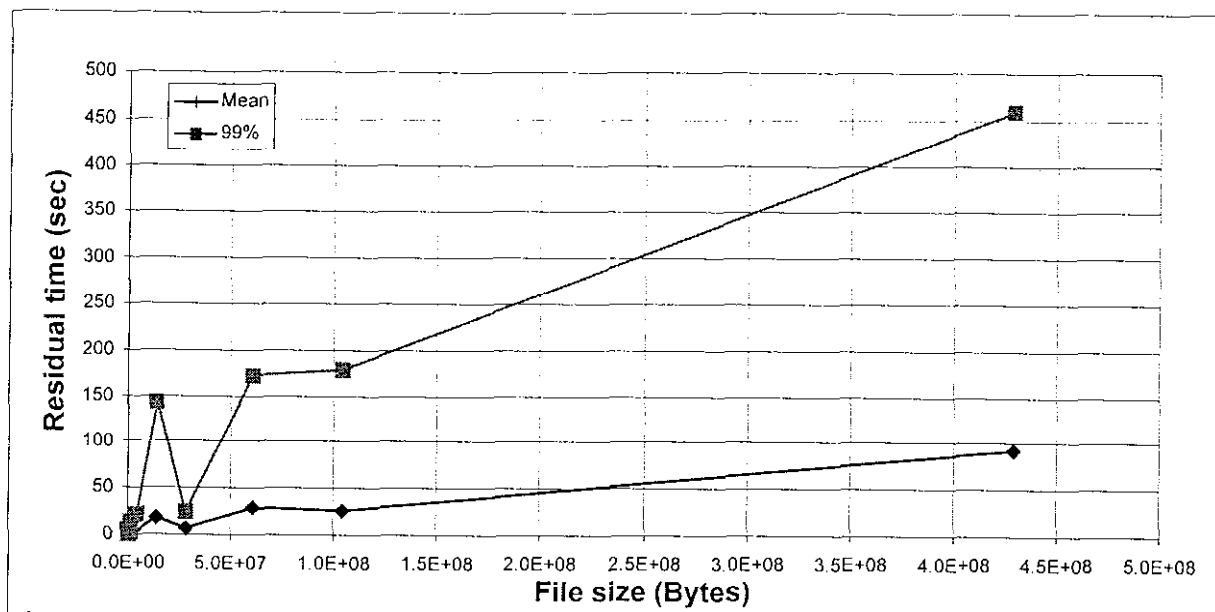


Figure 5.5 Mean and 99th percentile for blocks of 200 reads



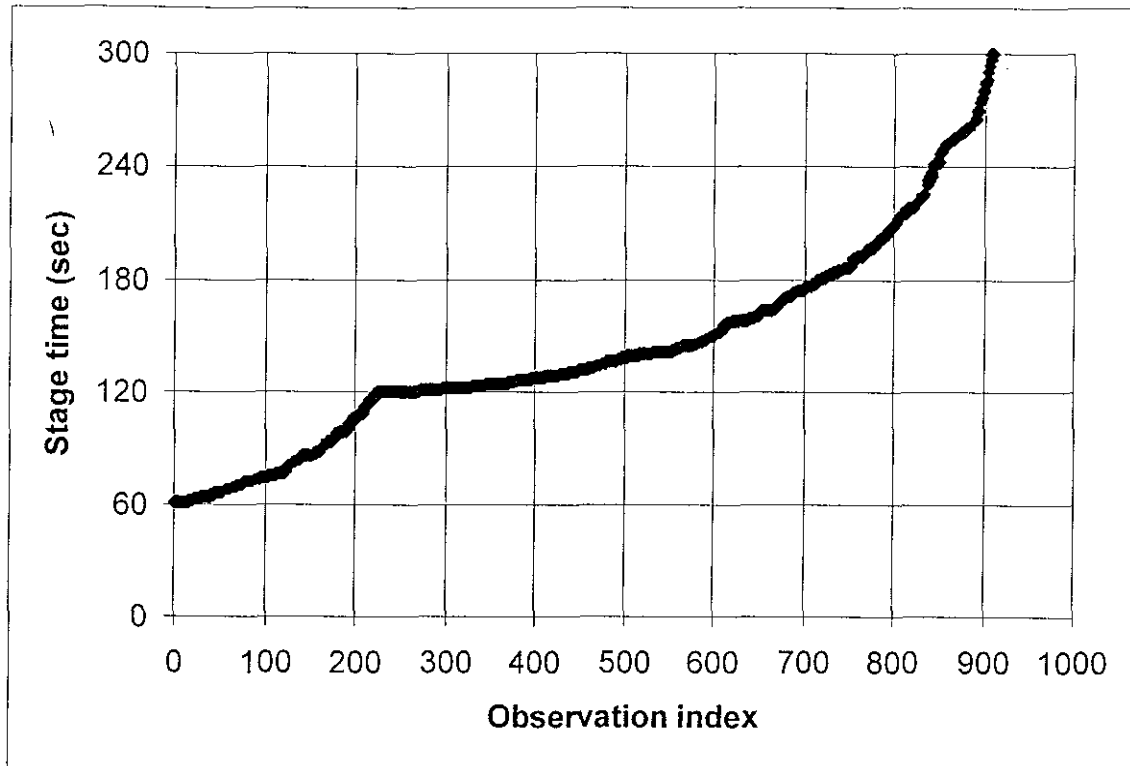
6.0 File staging times

The Unitree system transfers files between users and the disk cache. When the disk cache reaches its high water mark of 80%, files are prioritized and migrated from the disk cache to tape using a prioritization scheme that is based upon the time since the file was last accessed and the file size (priority = $M \cdot H^{1.5}$, where M is file size in MB and H is hours since last access). Files are migrated until the disk cache reaches its low water mark of 70%.

Occasionally, a file will be requested that is not in the disk cache. This will require retrieval and mounting of the appropriate tape. To estimate the frequency of tape mounts and the time required, the COND and STAT files from Endeavor file system were used. The formats of these files are shown in Attachment B. Note that the job "persona" field can be used to link the information in the COND and STAT files, and that the path description at the end of the COND file record have been changed to overwrite any sensitive information with sequential six digit numbers.

File staging times were extracted from the STAT files for May 1-14, 1998. The set of 996 data points was sorted by time and plotted, as shown below in Figure 6.1. Note that the minimum time required to stage a file is 60 seconds, and that there is a discontinuity in the curve at 120 seconds.

Figure 6.1 File stage times

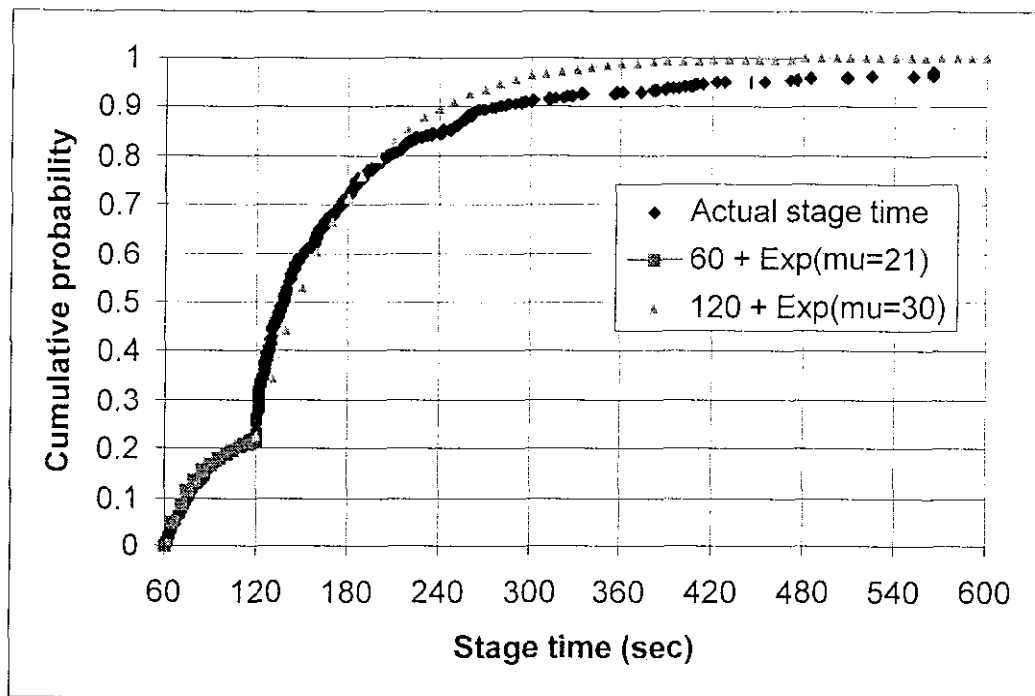


The time required to stage files can be represented by a pair of exponential probability distributions. The procedure for generating file stage times is as follows:

- 1) Draw random number from uniform distribution on $[0,1]$
- 2) If random number > 0.224 , go to step 6)
- 3) Draw random number from exponential distribution with mean = 21
- 4) Retain value from 3) if < 60 , repeat step 3) if > 60
- 5) Add 60 seconds to value realized in step 3), result is stage time, stop
- 6) Draw random number from exponential distribution with mean = 30
- 7) Add 120 seconds to value realized in 6), result is stage time, stop

This procedure generates file stage times that are representative of observed stage times. The distributions and observed data are compared in Figure 6.2.

Figure 6.2 Exponential distributions fit to file staging times



7.0 Summary and conclusions

The file transfer data suggest that contention for the 13 Unitree ports may be causing a wide variation in file transfer times. Transfer times for the same size file differed by as much as a factor of six for the 20,000 file transfers analyzed. Inherent latencies for writes and reads are 0.76 seconds and 0.20 seconds, respectively. Observed file transfer rates were 4.7 and 4.8 MB per second for writes and reads, respectively.

On average, contention for file transfer resources adds approximately 15 seconds to the file transfer time for a 10 MB file. In extreme cases, the worst 1% of transfers, contention for resources can add over 100 seconds to the file transfer time of a 10 MB file.

File staging takes from 60 to more than 600 seconds. A pair of exponential probability distributions fit the file staging times, with a break point at 120 seconds.

Attachment A – File Transfer Logs

Daily file transfer logs were provided by George Richmond for October 1, 1997 through April 9, 1997. The fields in the raw data files are described by the key below, where the two letter combination from below indicates the source and destination machines.

- a Various Sun Solaris workstations (one machine)
- b Meiko CS-2 (256 nodes)
- c Various Sun Solaris workstations (two machines)
- d DEC Alpha Server 8400 Model 5/440 (six machines)
- i Various IBM RS/6000 990s (four machines)
- j Cray J-90s (three machines)
- k IBM SP2 (256 nodes)
- l Various Sun Solaris workstations (one machine)
- r Various Sun Solaris workstations (one machine)
- s Storage (Unitree disk cache)
- w Various workstations outside of Livermore Computing (lots of machines)
- y Cray YMP (one machine)

Fields in a typical file:

```

/----- User number
/ /----- Date at end of transfer
/ / /----- Time at end of transfer
/ / /----- Source and destination
/ / / /----- Length of file in bytes (can be greater than 2^32)
/ / / / /----- Time to transfer file in seconds
005041 97/10/01 00:01:41 js 7700 0.425
001194 97/10/01 00:01:42 js 13166758 20.415
001065 97/10/01 00:05:30 ds 3622088 2.675
005080 97/10/01 00:05:41 ls 12299 0.641
005080 97/10/01 00:05:44 ls 2079 0.367
005080 97/10/01 00:05:48 ls 1895 1.904
005080 97/10/01 00:05:51 ls 1512 0.568
005080 97/10/01 00:05:54 ls 4851 0.442
005080 97/10/01 00:05:56 ls 1368 0.370
001065 97/10/01 00:05:58 ds 14352384 9.029
```

Attachment B COND and STAT files

COND file format:

005641 00:20:29 00:20:56 2360 103 35 put -d/usr/tmp/001314/001315/001316/...

Columns	Contents
1-6	User number
8-15	Job start time
17-24	Job completion time
26-30	Job persona
31-33	Session number (100 = session 0, 101 = session 1, etc.)
35-40	Job number within session
41-?	Job description
?-?	Job completion status

STAT file format:

98/05/05 00:20:56.383 2360 File transfer (j-s) 34713600 bytes in 25.115 seconds

Columns	Contents
1-8	Date
10-21	Time
23-27	Job persona
29-?	Variable

Other "stat" file lines look like:

98/05/05 09:54:31.012 3871 File failure (i-s) 1307 bytes in 0.731 seconds -192 (Ftp 1000)

98/05/05 08:35:01.060 3298 File staged in 180 seconds ^____^ Source and sink
error codes

Appendix B - Simulation Model and Analysis Report

1.0 Introduction

The Unitree Archive at Lawrence Livermore National Laboratory was selected as a single component of the system to model. This component was selected due to the availability of existing logs with which the model could be validated.

This document provides the basis for the simulation model design and an analysis of the model results compared to actuals.

2.0 Summary

A high level discrete event simulation model of the Unitree Archive was built based on knowledge obtained from Lawrence Livermore and concurrence on assumptions. Logs received from Lawrence Livermore were analyzed to verify some of these assumptions. A scenario was created based on the logs as input to the simulation model. The model provided results which differ from actual data, but differences can be explained. Further refinement of the model and assumptions will increase model fidelity.

3.0 System Components

Components of interest in this system are the Unitree Disk and Tape Servers and the network connectivity to clients. Figure 1 above provides the entire system layout. Only the Unitree portion of the system is modeled.

3.1 Network

Single user limited to 10 transfers

Single host limited to 25 transfers.

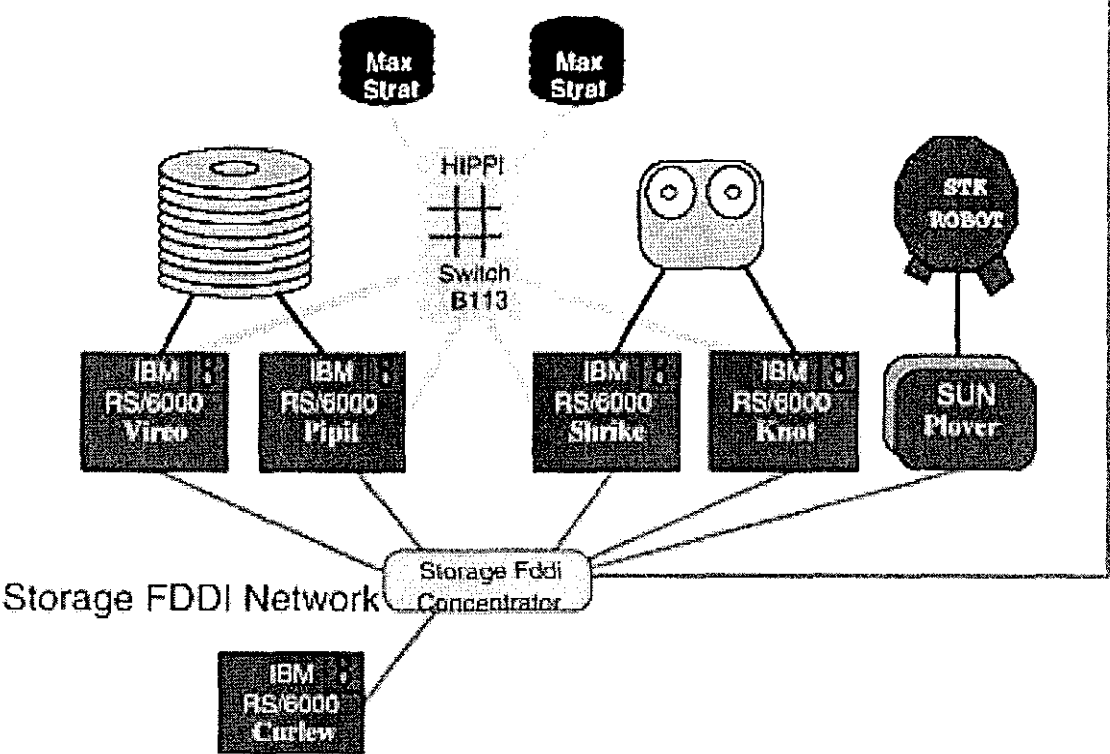
Single FDDI connection to Unitree disk server. Achieved rate is 7.5 MB/s (theoretical 12MB/s).

Average latency time for reads is 0.20 seconds and average latency time for writes is 0.76 seconds.

3.2 Unitree Disk Server

Server has 384 GB of disk storage across 6 SCSI chains. Each SCSI chain services 8 LUNs (which hold 8 GB). There are 4 logical volumes (LVs) per LUN each of 2 GB in size. Total LVs per SCSI chain is $4 \text{ LV/LUN} * 8 \text{ LUN/SCSI} = 32$. Total storage equals $6 * 8 * 8 \text{ GB} = 384 \text{ GB}$. The following sequence is used for allocating an LV for each incoming disk write (stores - user to Unitree server disk, and retrieves - tape to Unitree server disk): SCSI1 LV1, SCSI1 LV2, ..., SCSI1 LV32, SCSI2 LV1, SCSI2 LV2, ..., SCSI6 LV31, SCSI6 LV32, SCSI1 LV1,

Figure 1. Storage Sub-System Architecture Provided By LLNL



Files can be transferred to Unitree Server only if network ports are available. Table 1 shows qualifying ports based on file size. Model will always choose first available port for its file size (not round robin). Ports are needed only when transferring data between the Unitree server and a client.

Table 1: Network Ports Contention with Unitree

	0-4 KB	4 KB - 32 KB	32 KB - 512 KB	512 KB - 4 MB	4 MB - 256 MB	256 MB - 4 GB
Port 1	X					
Port 2	X					
Port 3	X	X				
Port 4	X	X				
Port 5	X	X	X			
Port 6		X	X			
Port 7		X	X	X		
Port 8			X	X		
Port 9			X	X	X	
Port 10				X	X	
Port 11				X	X	
Port 12					X	X
Port 13					X	X

Storage algorithm: 1. Create list of files that have been on disk for 1 hour or more and not yet stored. 2. Transfer these files to tape. 3. Sleep for 1/2 hour. 4. Repeat steps.

List of files for storage is ordered as all files on SCSI1 LV1, then all on SCSI1 LV2, and so on to SCSI6 LV32.

When a file exceeds 2GB in size, Unitree transparently uses 2 LVs. The maximum file size in Unitree is 4 GB.

Purge algorithm: A disk high water mark (80%) and low water mark (70%) are set. When disk fill hits high water mark, files are purged until disk fill below low water mark. Files must remain on disk for one hour beyond last read or write access. The order files are purged is based on weights. The weight of a file is calculated as $\text{Size_in_MB} * \text{Hours_Since_Last_Access}^{1.5}$. Size and hours are both rounded up to nearest integer.

3.3 Unitree Tape Server

Tape server has 3 SCSI chains. The bandwidth of each is 1 MB/s.

Only one (1) SCSI chain is used at a time for storing the list of files to tape. The SCSI chain selected depends on the tape currently being written to. Tapes are allocated incrementally based on cartridge number and cartridges are randomly distributed across the 5 silos.

Retrieves can occur on any of the three SCSI chains and at any time.

Data going to or coming from tape does not use disks which are resident to the Tape Server. Data flows from Unitree Disk Server disk over FDDI network and then through the Tape Server to tape. Data retrieval flows opposite path.

Each cartridge holds 1.2 GB of data.

4.0 Analysis of Log Data from Actual System

The NFT log files received contain transfer information between clients and Unitree disk only. Robot activity is logged in a separate file. NFT is performing the logging.

Logs contain transfers that do not include storage and transfers from storage to storage. Neither of these cases will be modeled.

Most files are cached. Smaller and recently accessed files stay around longer.

Retrieves are typically smaller files than writes. The large files that are written are rarely read back. There are no partial retrieves.

Table 2 shows a breakdown of transfer rates by host (code) and transfer direction (xs-store, sx-retrieve). There is considerable variation between hosts, and in some cases between stores versus retrieves. The disks on the different hosts may be a source of contention. Another possibility is the client process may be getting swapped out by the host due to its process loading.

Table 2: Observed Transfer Rates by Transfer Code

CODE	TRANSFER RATE				
	MEAN	STD	MIN	MAX	N
bs	0.84	0.66	0.00	2.70	330.00
ds	1.19	1.27	0.00	4.77	768.00
is	0.20	0.50	0.00	5.33	273.00
js	0.70	0.62	0.00	2.84	1338.00
ks	0.19	0.19	0.01	0.46	12.00
ls	0.01	0.01	0.00	0.02	6.00
sb	1.29	0.50	0.02	1.64	37.00
sd	1.93	1.21	0.10	3.94	33.00
si	0.27	0.68	0.00	4.00	102.00
sj	1.25	0.75	0.00	2.61	160.00
sw	0.32	0.14	0.00	0.72	162.00
sy	0.86	0.72	0.00	3.89	54.00
ws	0.64	0.63	0.00	4.95	851.00
ys	1.30	0.85	0.00	3.09	426.00

5.0 Model Design

5.1 Assumptions

1. Network host transfer limit of 25 has no impact on Unitree server since the server has a port limit of 13 simultaneous transfers. Since no impact on storage, do not model.
2. Assume SCSI chain rate on disk server is always less than or equal to the rate of an individual logical volume.
3. Tracking disk fill on an entire disk pool basis, not on a LV basis.
4. Modeling of archive will be simplified to a delay for the drive load. The detail of the archive robots and drives and related contention will not be modeled.
5. FDDI network is full duplex. For example, data flowing from client to disk can achieve 7.5 MB/s and at the same time data flowing from disk to client also achieves 7.5 MB/s.

5.2 Inputs

5.2.1 Scenario

The scenario that feeds the model contains the following fields: Time of Request, Transfer Code, User ID, Host ID, File Size, and File ID. The scenario was derived using two log files (stat and cond logs) obtained from LLNL for a period of 2 weeks beginning May 1. There was one file of each log type for each day.

The cond file contained fields for user ID, job start time, job completion time, job persona, session number, job number within session, and job description.

The stat file contained fields for date, time, job persona, and a text message. The text message was one of three types: 1 - transfer completed with file size, host id, and transfer time, 2 - transfer failed with file size, host, and failure time, and 3 - File staged and stage time. Staged means tape was accessed to get file.

To create the scenario, the stat and cond files were merged using the date and persona fields as matching keys. The date was assigned to cond files based on filename. Only records in cond file with a job description of "put" or "get" were maintained. For days where persona was not unique for that day (multiple occurrences of same persona), a set of personas were modified in the files to make them unique. Transfers that failed but were successful on a retry were kept in the scenario. Transfers that failed and had no successful retry were deleted. A couple cases with message "LEN failure" were also deleted.

Existing reusable submodels do not expect 0 byte file sizes. In order to force the simulation model to handle 0 byte file sizes, the file size was increased to 1 byte for these cases.

File ids were assigned to each record to represent unique filenames. File ids are used by disk submodel to determine if file is resident on disk or not. Each write (put) was issued a unique file id with the assumption that filenames were not written over (replaced). If the stat file specified staging took place, the corresponding job was also issued a unique id which would result in a tape access. All remaining reads (get) were assigned the id of the most recent write having the identical file size. If no match was found, then that file was written to disk at time 0 as part of the warmup time of the scenario.

5.2.2 Parameters

The following table contains a list of key parameters to the model.

Module	Description	Value
llnl	Maximum active transfers per user	10
llnl	Unitree Disk server SCSI rate (MB/S)	4.7
llnl	Unitree tape server SCSI rate (MB/S)	1
llnl	Latency time when initiating transfers for write (seconds)	0.76

Module	Description	Value
lInI	Latency time when initiating transfers for read (seconds)	0.20
lInI	Storage algorithm sleep time (seconds)	1800
lInI	Purge Algorithm: High water mark (%)	80
lInI	Purge Algorithm: Low water mark (%)	70
lInI	Purge Algorithm: Exponent in weight function	1.5
lInI	Tape load time (seconds)	60
lInI	Tape size (MB)	1200
transfers	Pseudo block size (MB)	1
transfers	Flow control block count	10
disk	Size of disk (MB)	384000
network	Effective transfer rate - FDDI (MB/s)	7.5

5.3 Data Flow

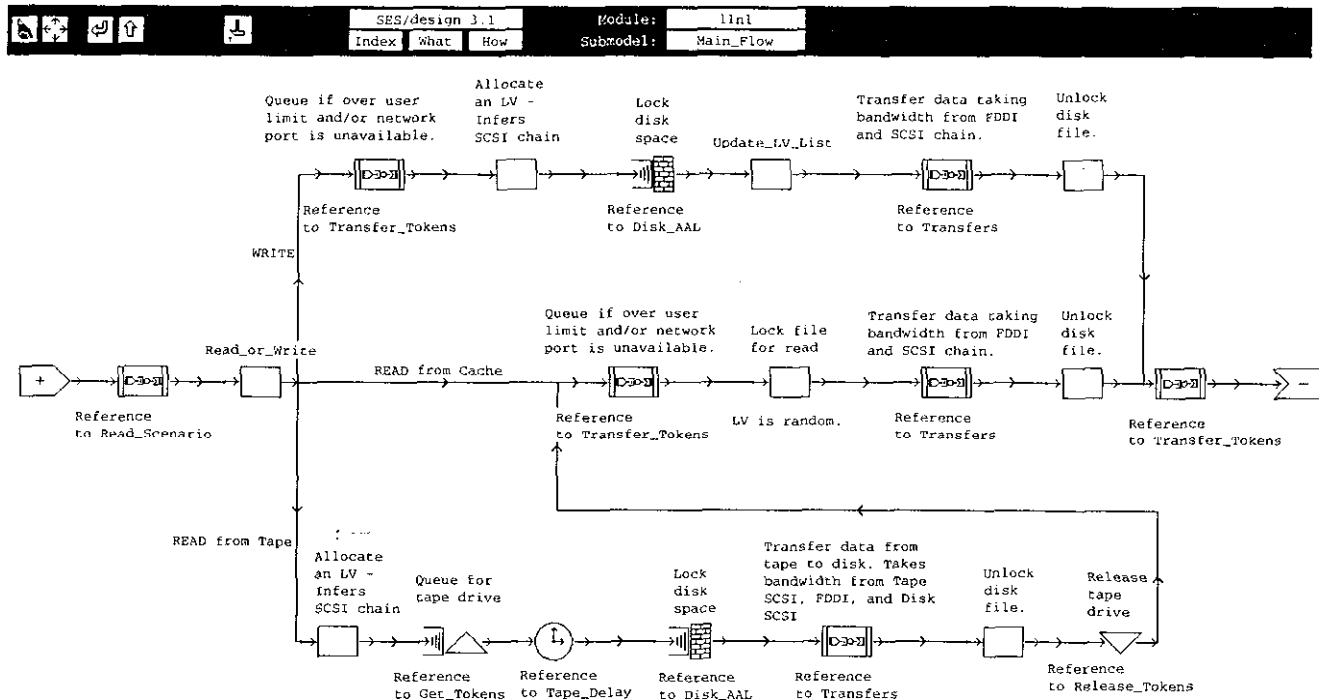
The simulation model will be built using six of the existing Open Architecture Model (OAM) reusable modules including the disk, network, resources, read_scenario, snapshot, and utilities modules. Two custom modules are created to represent the main flow of data and the transfer time between system components.

5.3.1 Main Flow

The main submodel begins by calling the Read_Scenario submodel for the generation of transaction based scenario file inputs. Each transaction will either be a write to storage or a read from storage.

The path for writes includes checking if a user is at his transfer limit and if a network port is available. A LV will be assigned in round robin fashion which then infers the SCSI chain to be used. Disk space is allocated and job will queue if disk space not available; however, disk space should be available if purge routine is working. Next is a delay for the transfer of data from client to Unitree Server disk. When the transfer is complete, a check on disk fill is performed and the purge routine is called if necessary. Lastly, decrement active transfers for user and release network port.

The path for reads depends on whether the file is cached or not. The first step for a read was to query the disk submodel and see if file was resident on disk. If file was not found in the disk list, then file was retrieved from tape. The input scenario specified the file id that was searched for on disk.



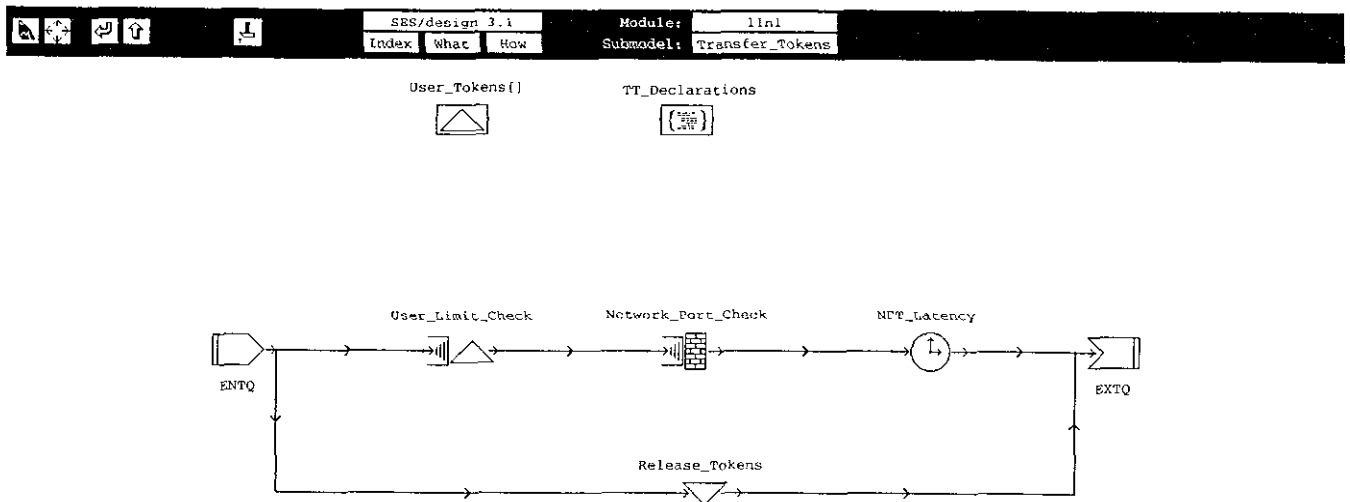
The path for a cached read is to check if user is over transfer limit and if network port is available. Next a lock is placed on the file so it cannot be deleted, then delay for the transfer from Unitree disk to the client. When transfer is complete, unlock the file and update the last access time. Lastly, decrement active transfers for user and release network port.

The data flow for a file coming from tape first includes a queue based on the tape SCSI chain and a delay for loading the tape drive. An LV is assigned in round robin fashion which infers the Unitree Disk SCSI chain to be used. A silo is randomly selected which infers the Unitree Tape SCSI to be used. (Silo 1 and 2 - SCSI 1, Silo 3 and 4 - SCSI 2, Silo 5 - SCSI 3) Disk space is allocated and then transfer of data from tape to disk via FDDI network begins. Once transfer completes, file is now on disk and path continues as a cached file would.

5.3.2 Transfer_Tokens Submodel

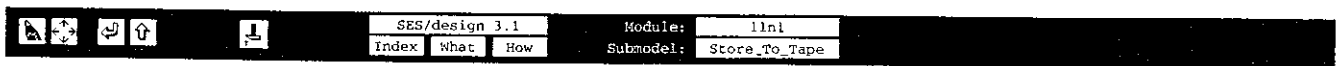
The Transfer_Tokens submodel supports the detail for checking if a user is at his transfer limit and if a network port is available. This submodel is called before and after a transfer between Unitree disk and a client. Before a transfer, a transaction will queue if tokens are not available. Once tokens are available an overhead latency due to NFT script is incurred, then transaction returns to calling submodel to execute the actual transfer.

After a transfer, the submodel is called to release tokens for subsequent transactions.

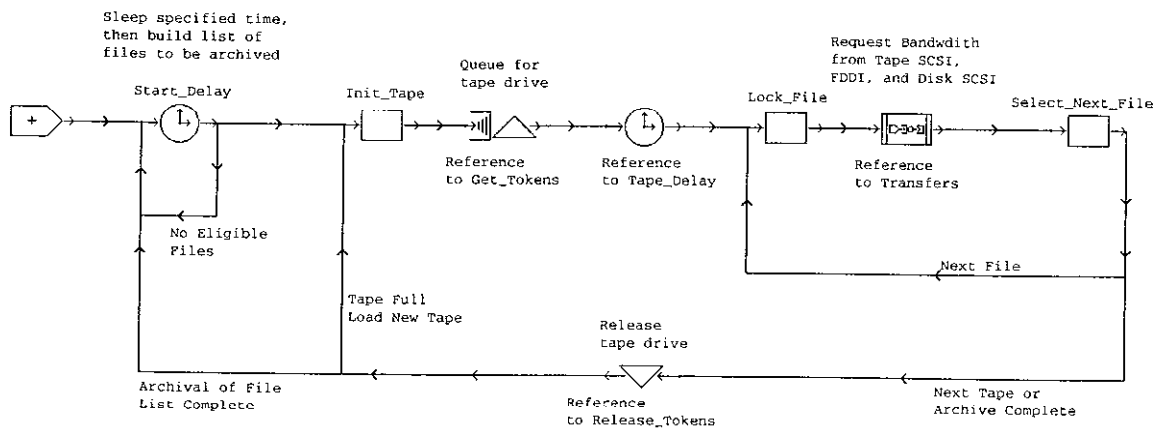


5.3.3 Store to Tape

The Store_To_Tape submodel represents the background process of archiving files to tape. A single process begins with a delay for a parameterized time (1/2 hour). At the end of this sleep time a list of files eligible for storage (files on disk for 1 hour or longer and not yet on tape) is created. The tape SCSI is determined by randomly selecting a silo. The job will queue for a tape drive based on tape SCSI chain. When drive is available, a delay is incurred to load a tape. The LV on which the file resides is used to assign the disk SCSI chain. Next is a delay for the transfer of a file from disk to tape via FDDI network. The transfer delay is repeated until the tape is filled, then another tape load delay is incurred. Once all files have been stored, the process begins again with the sleep delay.



Submodel_Declarations

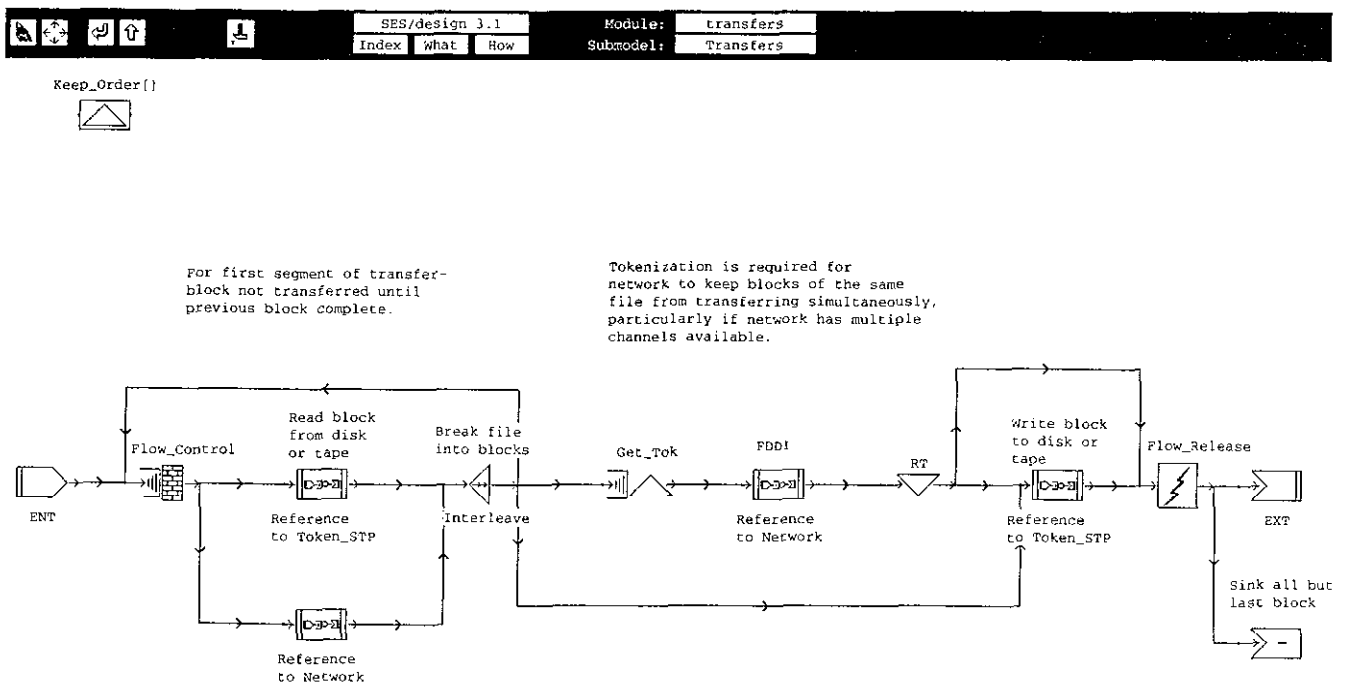


5.3.4 Transfers Submodel

The Transfers submodel models the delay incurred when transferring data between different components of the system. Contention of resources is modeled by breaking a file into pseudo-blocks and interleaving the pseudo-blocks when multiple transfers use the same resources.

The resources used depend on the transfer type. Transfers between client and disk use the network resource and a resource representing a disk SCSI chain. Transfers between tape and disk use a resource representing a tape SCSI chain, a resource representing a disk SCSI chain and the network resource.

Since there may be considerable differences in rates of the three resource types, a flow control exists to limit the number of pseudo-blocks transferring at a given time for a particular file. This number is a parameter.



6. Model Results

Detail model statistics can be found in the Appendix.

The first noticeable statistic is the delays due to user limits and network ports. Both of these queues contained jobs queueing up to 1 and 2 hours.

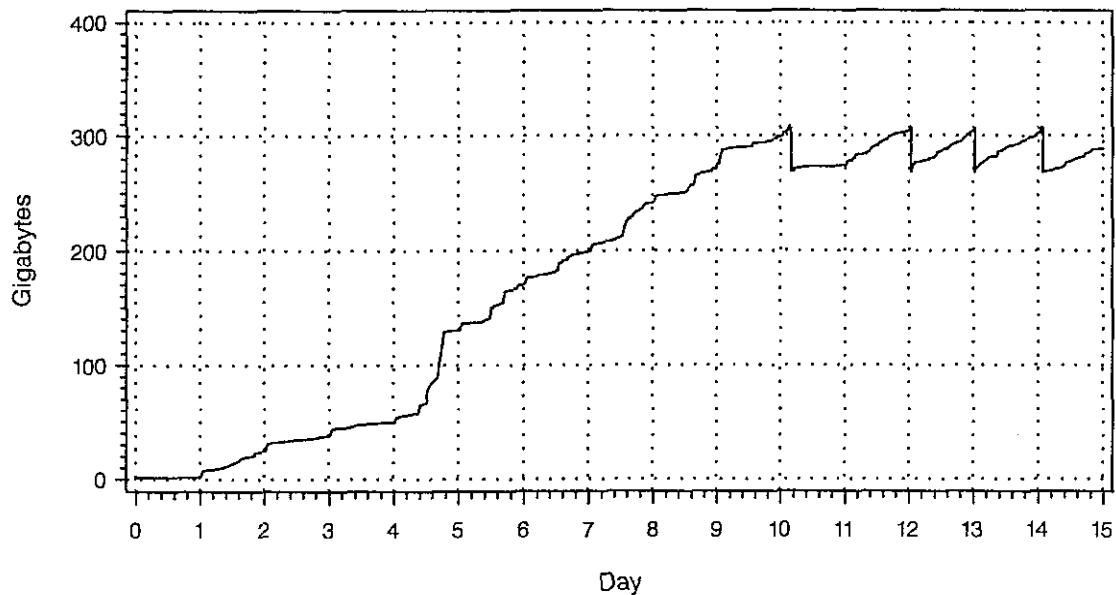
Tape accesses account for about one-third of the gets. Although the average number of tape accesses is only about 3 per hour, there are periods when all 20 tape drives are active. Given peak loadings on tape allocation and user and network queues, it appears that some users tend to transfer files in bulk (many at once). The nature of the bursty file requests by users implies that conducting future modeling runs with different user loads should also include a bursty nature of user requests. Assuming a steady stream of requests may not yield reasonable results.

As expected, the network is not a bottleneck. The average utilization of the network was 5%.

For this modelling exercise, the disk does not reach its high water mark until the second week. The purge algorithm was executed only 4 times. The following graph shows disk usage over the two week period. Not until day 9 did the model (based on disk fill) reach steady state. This data implies that future model runs should include an additional 10 days in the scenario for warmup time.

Figure 2 - Disk Fill

Disk Fill for Duration of Model Run



A log containing the transfer time and tape access time (if tape needed) for each job was created during the simulation model run. The model log was compared to the actual data resulting in the following tables.

Table 3 shows that the model used the same number of jobs and file sizes as the actual data. One difference was modelling the 0 byte files as 1 byte files.

Table 3: File Size Comparison (Bytes)

		MEAN	STD	MIN	MAX	N
get	Actual	5858930.08	22627776.94	12	462234843	3061
	Model	5858930.08	22627776.94	12	462234843	3061
put	Actual	17029038.69	81959836.77	0	2275901440	24441
	Model	17029038.69	81959836.77	1	2275901440	24441

Tables 4 and 5 show statistics for the file transfers. The model results are showing smaller transfer times. One source of difference is the potential for a host to swap out the transfer process, which was not modeled, thus lengthening the transfer time.

Table 4: FTP Delay Time Statistics (Seconds)

		Mean	STD	Minimum	Maximum	N
get	Actual	4.10	14.97	0.08	308.70	3061
	Model	2.96	10.21	0.20	259.19	3061
put	Actual	11.22	40.70	0.37	1403.83	24441
	Model	8.92	33.66	0.76	711.99	24441

The lower percentiles for the model data have less variation compared to actual data. The reason for this is the constant overhead rate that was modeled based on regression analysis of the actual data.

Table 5: FTP Delay Time Percentiles (Seconds)

		1st Percentile	5th Percentile	10th Percentile	Median	90th Percentile	95th Percentile	99th Percentile
get	Actual	0.097	0.130	0.140	0.739	6.017	17.839	64.781
	Model	0.200	0.201	0.204	0.507	6.311	11.022	49.329
put	Actual	0.527	0.614	0.829	3.258	18.655	34.267	174.198
	Model	0.760	0.763	0.765	2.106	14.644	24.135	131.574

Tables 6 and 7 reflect statistics on the tape staging time. One observation from the actual data was removed from analysis as an extreme outlier. Some differences were expected here due to the simplistic approach of modeling robot and drive overhead as a single constant delay. More details could be modeled which would require more knowledge of the tape and drive system. Information like when tapes are dismounted, positioning speeds, drive load time, robot move time. Assumptions would be made concerning location of files on tapes. Some files may be on same tape, but only one file can be read at a time. The single constant delay was set to 60 which matched the minimum tape delay from actual data. Using a delay of 120 seconds matching the median of actual data would improve the correlation in the upper percentiles between actual and raw data.

Table 6: Tape Delay Time Statistics (Seconds)

		Mean	STD	Minimum	Maximum	N
get	Actual	196.2	260.28	61.00	3187.00	999
	Model	110.20	90.87	60.00	833.29	1003

Table 7: Tape Delay Time Percentiles (Seconds)

		1st Percentile	5th Percentile	10th Percentile	Median	90th Percentile	95th Percentile	99th Percentile
get	Actual	61.000	66.000	74.000	138.000	281.000	444.000	1586.000
	Model	60.000	60.006	60.019	70.775	183.555	282.448	526.050

The total times represented in Tables 8 and 9 also show differences between the model and actual. It is evident that the actual data reflects points of contention that were not included in the model. These points of contention have not been clearly identified. One possible difference is due to the contention created by file transfers that failed and were not modeled. Another difference could be due to the assumption of which files were being read (most recent matching file size).

The actual data included severe outliers having times in excess of 86300 seconds. Further analysis showed that the cond files from which the total time is calculated contained invalid data - the start times were greater than the end times. These observations were removed from the statistics gathered on the actual data.

Another point of investigation may be the user limit queue and network port queue. The model serialized theses queues. Results could be different if modeling a single queue which released first job found matching both user and port criteria.

Table 8: Total Request Time Statistics (Seconds)

		Mean	STD	Minimum	Maximum	N
get	Actual	182.34	424.00	0.00	10792.00	3058
	Model	40.14	79.55	0.20	869.16	3061
put	Actual	104.62	247.66	0.00	6418.00	24409
	Model	31.03	256.80	0.76	8037.65	24441

Table 9: Total Request Time Percentiles (Seconds)

		1st Percentile	5th Percentile	10th Percentile	Median	90th Percentile	95th Percentile	99th Percentile
get	Actual	1.000	2.000	2.000	51.000	517.00	919.000	2048.000
	Model	0.200	0.204	0.204	1.504	119.739	165.323	404.271
put	Actual	1.000	2.000	3.000	17.000	328.000	577.000	832.000
	Model	0.760	0.764	0.766	2.747	52.004	110.387	297.856

7. Model Usage

Once a simulation model has been developed and validated, it is ready to be applied toward making design decisions. Design decisions can be based on simulation results obtained through the varying of parameters to the model or through the changes in input loading to the model.

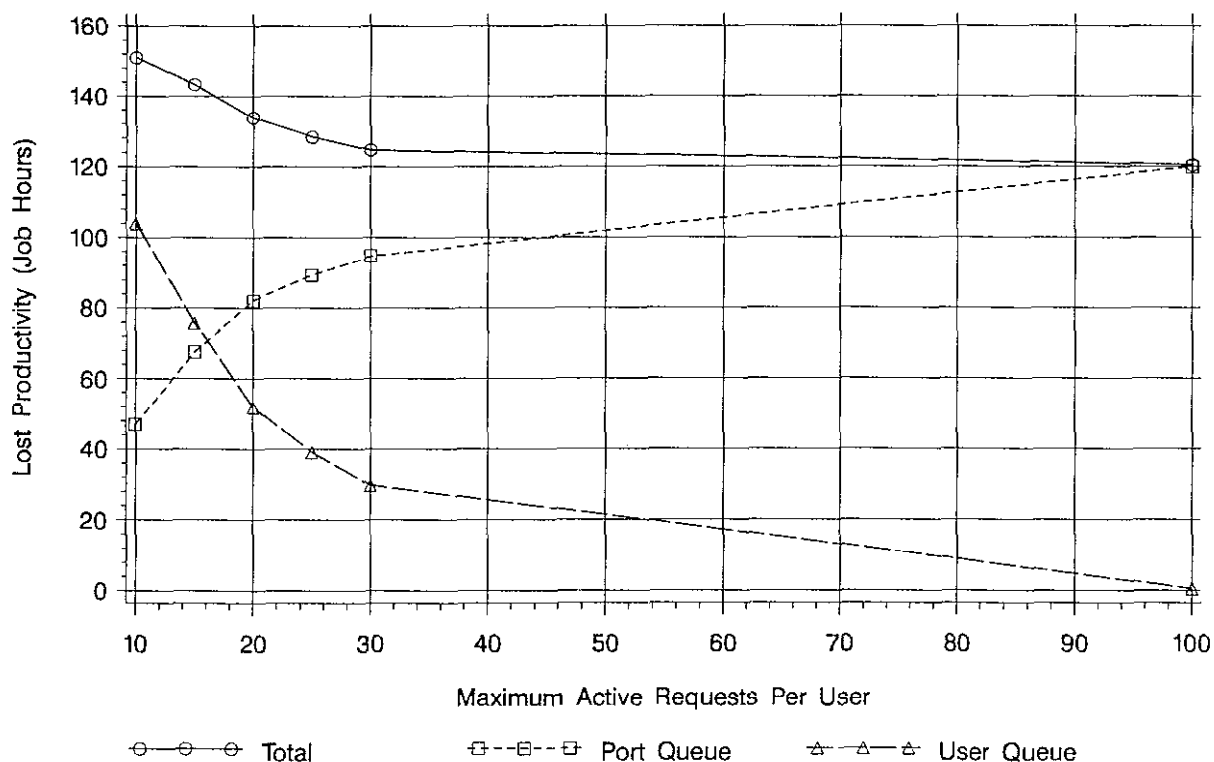
An exercise performed with the LLNL model was to quantify the productivity impact by changing the limit of active user requests. The following table contains results from a series of six simulation runs with varying user limits. The total number of read requests that completed was 3061 and the total number of write requests that completed was 24441.

Table 10: User Limit Statistics

User Limit	10	15	20	25	30	100
User Limit Mean Queue Time	78.9484	79.6763	72.4854	68.8343	63.7842	14.0321
Network Port Mean Queue Time	30.8128	43.0581	51.6697	55.9193	59.1574	73.8711
User Limit Count	4742	3421	2575	2041	1683	148
Network Port Count	5491	5654	5719	5752	5772	5841

Lost productivity is defined as the amount of time which a job was idle due to queueing. Lost productivity due to user limit can be calculated by multiplying the mean queue time for user queue and jobs that queued. Lost productivity due to network port limit can be calculated by multiplying the mean queue time for port and number of jobs that queued for a port. The following graph shows how increasing the user limit can increase productivity.

Figure 3 - Lost Productivity



The amount of lost productivity decreased as user limit was increased. These results lead to the question of why have a user limit. Discussions with NSL Unitree domain experts lead to finding that the Unitree system will fail if number of active

jobs per user was increased. A productivity improvement could be made if software or hardware changes resulting in a higher user limit could be made.

Similar studies could be made by performing trade studies on other parameters to the model.

Design decisions can also be made by varying the input load to the model. The input for this model was a scenario created from two weeks of actual log data. Longer scenarios based on actual data could be studied. Scenarios based on forecasted loadings could also be created and then used to determine when a system may break.

8. Sample Model Output

	Mean	Standard Deviation	Minimum	Maximum	Count or Last (LV)
	----	-----	-----	-----	-----
Main Submodel Statistics:					
Time queued due to user limit	78.9484	440.0954	0.0100	6715.7734	4742
Time queued due to network port limit	30.8128	183.0612	0.0000	3538.0938	5491
Total Time of Store Process (Min)	18.8150	44.0326	0.0000	508.6361	413
Files Per Store Process	59.1065	97.2636	0.0000	705.0000	413
Total Data Per Store Process (MB)	1006.8642	2451.6381	0.0000	28274.6988	413
Total Tapes Per Store Process	1.6610	2.5479	0.0000	32.0000	413
Files Per Tape	37.8945	50.8797	1.0000	283.0000	275
Total Data Per Tape (MB)	983.4823	376.4744	0.0247	2275.9014	275
Tapes in Load Process	0.0838	0.6678	0.0000	20.0000	0.0000 LV
Purge Alg: Files Purged	70.5000	39.0768	39.0000	126.0000	4
Purge Alg: Blocks Purged	39518.0000	290.3458	39162.0000	39863.0000	4
Purge Alg: Time since last purge (sec)	304001.8131	383942.8376	85335.5847	877505.8759	4
Transfers Submodel Statistics:					
Active transfers in system	0.5518	0.9984	0.0000	24.0000	0.0000 LV
Active ftp gets in system	0.0070	0.1329	0.0000	7.0000	0.0000 LV
Active ftp puts in system	0.1650	0.7195	0.0000	13.0000	0.0000 LV
Active tape writes in system	0.3514	0.4774	0.0000	1.0000	0.0000 LV
Active tape reads in system	0.0285	0.4214	0.0000	20.0000	0.0000 LV
Transfer times for ftp put (sec)	8.1644	33.6622	0.0000	711.2257	24441
Transfer times for ftp get (sec)	2.7578	10.2066	0.0000	258.9888	3061
Transfer times for tape writes (sec)	17.4110	83.1677	0.0000	2398.7715	24411
Transfer times for tape reads (sec)	34.3431	85.8927	0.0000	773.2854	1003
Transfer sizes for ftp put (MB)	17.0290	81.9598	0.0000	2275.9014	24441
Transfer sizes for ftp get (MB)	5.8589	22.6278	0.0000	462.2348	3061
Transfer sizes for tape writes (MB)	17.0347	82.0088	0.0000	2275.9014	24411
Transfer sizes for tape reads (MB)	12.4551	36.2936	0.0000	462.2348	1003
Network Submodel:					
network[0]					
channel[0] utilization (%)	4.9383	21.6666	0.0000	100.0000	0.0000 LV
network[1]					
channel[0] utilization (%)	4.9967	21.7876	0.0000	100.0000	0.0000 LV
Resources Submodel:					
Drive Queue 1[9]					
Token utilization	0.1862	0.5248	0.0000	8.0000	0.0000 LV
Allocation queue (entries)	0.0050	0.1952	0.0000	14.0000	0.0000 LV
Allocation response (sec)	8.5488	26.0570	0.0000	170.0869	711
Drive Queue 2[10]					
Token utilization	0.1872	0.5201	0.0000	8.0000	0.0000 LV
Allocation queue (entries)	0.0055	0.1999	0.0000	12.0000	0.0000 LV
Allocation response (sec)	10.0985	28.5935	0.0000	184.5633	659
Drive Queue 3[11]					
Token utilization	0.0903	0.3116	0.0000	4.0000	0.0000 LV
Allocation queue (entries)	0.0027	0.1333	0.0000	13.0000	0.0000 LV
Allocation response (sec)	10.1007	33.0580	0.0000	228.4576	319
Disk Statistics					
Fill level in MB	185497.9628	99429.0934	1934.7243	297409.9351	272717.1674 LV
Fill level in MB blocks	193311.1235	103978.1018	2036.0000	308519.0000	288909.0000 LV